THE BINARY GENETIC ALGORITHM

DR H.K. LAM

Office S2.14, Strand Building, Stra

Email: hak-keung.lam@

https://nms.kcl.ac

Nature-Inspired Learning Algorithms (7CCSMBIM)

Outline

KING'S College LONDON
University of London

Aims

- To understand the process of the binary genetic algorithms
- To apply the binary genetic algorithm to optimisation problems.
- To k

Objectiv

- To study how the binary genetic algorithm works in d
- To consider a number of applications and formula

Problem Statement:

- **Task**: minimisation of a cost function
- Gradient information is not required
- Fun

Difficulti

- non-convexity
- multi-modality
- non-smoothness
- discontinuity
- dimensionality

King's College London
University of London

The Genetic Algorithm

- Genetic algorithm (GA) is a technique to solve problems which need optimisation.
- GA is
- GA is

- History of GA
  - Evolutionary Computing evolved in the 1960's
  - GA were proposed by John Hollan in the middle of 1

Darwin's Theory of Evolution [1]

- An offspring has many of the characteristics of its parents, which implies that the population is stable.
- The d from

- Only a small percentage of the offspring produced s
- Which of the offspring survive depends on their fitne

[1] Sue Ellen Haupt, ValliappaLakshmanan, CarenMarzban, AntonelloPasini, and John K. Williams. Environmental Science Models and Artificial Intelligence. Artificial Intelligence Methods in the Environmental Sciences. Springer Science (3-14, 103-126), 2009.

KING'S College LONDON
University of London

The Binary Genetic Algorithm

- Biological Metaphor - Natural Selection
    - Genetics and Evolution - gene, chromosome, allele, genotype, phenotype,

- Co
    - ting mutation, offspring and convergence, · ·

Advantages:

- Optimise with continuous or discrete variables.
- Derivative information is not required.
- Able

- Opti
- Is less likely trapped in local optimum.
- Tends to search for global optimum.

KING'S College LONDON
University of London

- N_{var} : number of decision variables of the chromosome
- N_{bits} : total number of bits of the chromosome
- N_{pop} : population size
- N_{pop}
- X_{rate}
- N_{kee}
- $N_{pop} - N_{keep}$: number of chromosomes to be discarded
- x_{lo}: lower bound of variable x
- x_{hi}: upper bound of variable x
- P_n : probability of the n^{th} chromosome in the mating pool of N_{keep} to be chosen
- c_n : cost of the n^{th} chromosome
- C_n : normalised cost of the n^{th} chromosome
- μ : mutation rate (or probability of mutation)

Assignment Project Exam Help

https://eduassistpro.github.i

Add WeChat edu_assist_pr

Initial population with random members

Reproduction

Mutation

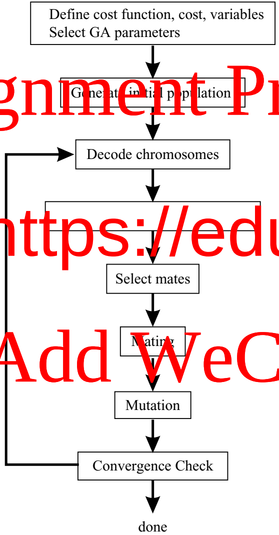


Figure 1: Flowchart of a binary genetic algorithm.

Initial population with random members

Reproduction

Mutation

Binary number conversion:

Example: Convert 25.3125 to binary

The integer part: 25

The fractional part: 0.3125

$\boxed{0}$

$\boxed{1}$

$6/2 \longrightarrow \boxed{0}$

$3/2 \longrightarrow \boxed{1}$

$1/2 \longrightarrow \boxed{1}$

$$25.3125_{10} = 11001.0101_2$$

Binary to Decimal:

$$(\mathbf{1}\times 2^4 + \mathbf{1}\times 2^3 + \mathbf{0}\times 2^2 + \mathbf{0}\times 2^1 + \mathbf{1}\times 2^0).(\mathbf{0}\times 2^{-1} + \mathbf{1}\times 2^{-2} + \mathbf{0}\times 2^{-3} + \mathbf{1}\times 2^{-4})$$

Given a number $x \in [x_{lo}, x_{hi}]$, how many bits (m) are required to achieve precision of d decimal places?

Example

$\frac{100-25}{10^{-2}} \leq \quad - \quad \Rightarrow \quad \leq \quad \Rightarrow \quad = \quad .$

$$0000000000000 \rightarrow 25 + 0 \times \frac{100-25}{2^{13}-1} = 25$$
$$0000000000001 \rightarrow 25 + 1 \times \frac{100-25}{2^{13}-1} = 25.0092$$
$$0000000000010 \rightarrow 25 + 2 \times \frac{100-25}{2^{13}-1} = 25.0183$$

$$\boxed{\textbf{Decoding: } x = x_{lo} + decimal(1001\cdots001_2)\frac{x_{hi}-x_{lo}}{2^m-1}}$$

- The optimisation/decision variables are represented by *chromosome*.

$$chromosome = [p_1, p_2, \cdots, p_{N_{var}}]$$

- Eac

- Tot

- The cost is evaluated by a cost (fitness) function.

$$cost = f(chromosome)$$

- Genotype: The bit string representation of the chromosome
- Phenotype: The decision variables. The genotype can be mapped to phenotype through decoding or vice versa through encoding.
- Allele: the value of a single bit in the chromosome

Example: Consider an optimisation problem with decision variables of $p_1, p_2, \cdots,$ $p_{N_{var}}$.

- Phenotype: $[p_1, p_2, \cdots, p_{N_{var}}]$
- Wh

Chr

$\underbrace{}_{gene_1(p_1)} \underbrace{}_{gene_2(p_2)}$

Genotype: 1100110011 0111110000 $\cdots$

Allele: the allele of the first bit from the left is '1'.

- The GA starts with a group of chromosomes known as the population.

- The population has N_{pop}.
- A population represented by a N N matrix filled with random 0s and 1s.

- **Pur**

evolved to improve their quality in each generation

Example: A cost function: $cost = f(x,y)$ with 7 bits in each gene.

$$chromosome = \Big[\underbrace{1100011}_{x}\ \underbrace{0011001}_{y}\Big]$$

[illegible]	Co
11100101100100	11872
00110010001100	[illegible]
00101111001000	[illegible]
11001111111011	[illegible]
01000101111011	-12097
11101100000001	-12588
01001101110011	-11860

Table 1: Example initial population

King's College London
University of London

- **Two approaches**: X_{rate} and Thresholding.

- **Purposes**: Determine who should survive and who should die. The stronger ones will survive and the weaker ones will die, i.e., "Survival of the fittest" from Dar

Natural Selection using X_{rate}

- Survival of the fittest: Only the best are selected to survive.
- Natural selection occurs each generation or iteration of the algorithm.
- The s

- The n

$N_{keep} = X_{rate} N_{pop}$.

- The **top** N_{keep} will be kept in each generation.
- The **bottom** $N_{pop} - N_{keep}$ chromosomes w

new offspring.

Example: $N_{pop} = 8$ and $X_{rate} = 50\%$. $N_{keep} = X_{rate}N_{pop} = 0.5 \times 8 = 4$.

Chromosome	Cost
00110010001100	13477
01000101111011	
11100101100100	
01001101110011	
11001111111011	–

Table 2: Ranked population. The upper four will be kept and the lower four will be discarded and replaced.

Natural Selection using Thresholding:

- All chromosomes that have a cost **lower** than a pre-defined threshold survive.
- Chromosomes with a cost **higher** than the threshold will be discarded and repl
- If no c

- The threshold can be changed in each generation.

Advantage over X_{rate} natural selection:

- Less computationally expensive as population d

Selection

Selection:

- **Four approaches:**

  1) Pairing from top to bottom
  2)
  3)
     3.2) Cost weighting
  4) Tournament selection

- **Purposes:** Determine who should reproduce o

Selection: Two chromosomes are selected from the mating pool of N_{Keep} chromosomes to produce two new offspring. Selection will take place until $N_{pop} - N_{keep}$ offspring are born to replace the discarded chromosomes.

1) **Pair**

   chro ted for mati

   Property: Simple and easy to implement.

2) **Random pairing:** A uniform random number g chromosomes.

   Property: All chromosomes (in the mating pool of N_{Keep}) have chance to mate. Introduce diversity to the population resulting in higher chance of producing offspring of quality.

3) **Weighted Random Paring (roulette wheel weighting):** The probabilities assigned to the chromosomes in the mating pool are inversely proportional to their cost.

- **Pro**

  mati

  of mating.

- **Two techniques:** Rank weighting and cost we

3.1) **Rank weighting (roulette wheel weighting):**

$$P_n = \frac{N_{keep} - n + 1}{\sum_{n=1}^{N_{keep}} n}$$

whe [illegible] N_{keep} to be

cho [illegible]

Example

n	Chromosome	Cost		
1	00110010001100	[illegible]	[illegible]	[illegible]
2	11101100000001	$-$	0.3	0.7
3	00101111001000	-12363	0.2	0.9
4	00101111000110	-12359	0.1	1.0

Table 3: Probability table for rank weighting.

3.1) **Rank weighting (roulette wheel weighting):**

Properties:

- It is problem independent.
- .
- tionally

3.2) **Cost weighting (roulette wheel weighting):**

Normalised Cost: $C_n = c_n - c_{N_{keep}+1}$

Probability: $\frac{C_n}{\text{[illegible]}}$

Example

n	Chromosome	$C_n = c_n - c_{N_k}$ [illegible]	P_n [illegible]	$\sum^{n} P_n$ [illegible]
1	00110010001100	$-13477 + 12$ [illegible]	[illegible]	[illegible]
2	11101100000001	$-12588 + 12$ [illegible]	[illegible]	[illegible]
3	00101111001000	$-12363 + 12097 = -266$	0.111	0.891
4	00101111000110	$-12359 + 12097 = -262$	0.109	1.000

Table 4: Probability table for cost weighting.

3.2) **Cost weighting (roulette wheel weighting):**

Properties:

- It is cost function dependent.
-

-

  approximately the same cost.
- The probabilities have to be calculated each gen ... nally expensive.

Remark for Normalisation: Different scaling functions can be used. For example, a more general form of scaling function could be

where a

In our example, we choose $a = 1$ and $b = -c$

$$C_n = c_n - c_{N_{keep+1}}$$

4) **Tournament selection:**

- Randomly pick a small subset of chromosomes (two or three) from the mating pool in the N_{keep}, and the chromosome with the lowest cost in the subset

Pro

-
- It works best for larger population sizes because s time-consuming for large populations $\Rightarrow$

- Chromosomes of good quality (with lower cost) h chosen.

Crossover:

- **Two approaches:**

1) Single-point crossover
2)
3)

- **Pur**

process (by exchanging information).

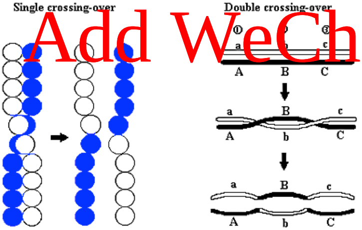
1) **Single-point crossover**:

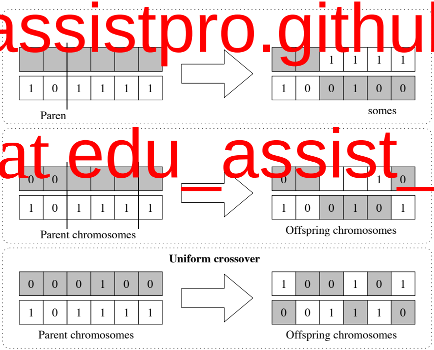
Step 1: A crossover point is randomly selected between the first and last bits of the parents' chromosomes.

Step 2: Generate two offspring by swapping the chromosomes from the crossover point between two parents.

Step 3: Replace any two chromosomes to be discarded in the pool of $N_{pop} - N_{keep}$ in the population.

Step 4: ced.





1) **Single-point crossover**:

Example: $N_{keep} = 4$

Chromosome	Family	Binary String
6	$offspring_2$	
3	ma(2)	
4	pa(2)	
7	$offspring_3$	*0010111100*0110
8	$offspring_4$	0010111100*1000*

Table 5: Pairing and mating process of single-point crossover.

2) **Double-point crossover**: The segments in between two randomly generated crossover points are swapped between parents.

3) **Uniform crossover**: Bits are randomly chosen for swapping between parents.

Mutations:

- **Purposes:** Random mutations alter a certain percentage of the bits in the list of chromosomes. It allows the GA to explore a cost surface by introducing new information.

Mutation pr

Step 1: Choos

Step 2: Deter

$$\#mutation = \mu(N_{pop} - 1)N_{bits} \text{ (elitism is impl}$$

Step 3: Flip the chosen bits.

Remark: If elitism is NOT implemented,

$$\#mutation = \mu N_{pop} N_{bits}.$$

- **Mutations**:

Example: $N_{keep} = 4$, $\mu = 0.2$, $\#mutation = 0.2(8-1)14 = 19.6 \approx 20$

Population after Mating	Population after Mutations	New Cost
		477
		588
		415
00101111000110	0000101	
00101100000001	0010110	
11101111001000	1111011	
00101111000110	00100111101000	−12716
00101111001000	00110111001000	−12103

Table 6: Mutating the population.

Chromosome	Cost
00001011000111	-13482
00110010001100	-13477
00101111010000	
11110111010010	
00110111001000	

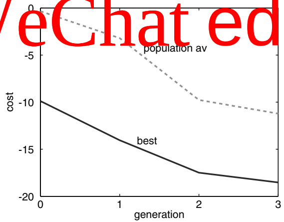
Table 7: New ranked population at the start of the second generation.

King's College London
University of London

Stopping Criteria:

- Whether an acceptance solution is reached.
- A set number of iterations is exceeded.
- No c
- No c

- Population statistics on mean and minimum cost.



Question: Find the maximum percentage of the possible solution being searched after 3 generations

Total num

maximu

$\underbrace{8}_{\text{initial population}} + \underbrace{7}_{\substack{\text{max cost evaluations per generation} \\ \text{(elitism is implemented)}}} \times$ gene

$\frac{29}{128 \times 128} \times 100 = 0.18\%$ of the solution space has been searched.

Statistics

- mean, standard deviation, the worst and the best of the cost of multiple runs

Converg

Benchm

- Functions of different properties

Function 1:

$$f_1(\mathbf{x}) = \sum_{i=1}^{n} x_i^2, -5.12 \leq \quad _i \leq$$

minimum: $\mathbf{x}^* = 0, f_1(\mathbf{x}^*) = 0$

Function 2:

$$f_2(\mathbf{x}) = \sum_{i=1}^{n-1} \left(100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2 \right), -2.048 \leq x_i \leq 2.048$$

minimum:

Function

$$f_3(\mathbf{x}) = 6n + \sum_{i=1} floor(x_i), -$$

minimum: $x_i^* = [5, 5.12], f_3(\mathbf{x}^*) = 0$

Function 4:

$$f_4(\mathbf{x}) = \sum_{i=1}^{n} i x_i^4 + Gauss(0,1), \quad -1.28 \le x_i \le 1.28$$

minimum:

Function

$$f_5(\mathbf{x}) = \frac{1}{k} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2} (x_i - a_{ij})^6},$$

maximum: $x_i^* = -32, f_5(\mathbf{x}^*) \approx 1$ with $k = 500$

$$a_{ij} = \begin{bmatrix} -32 & -16 & 0 & 16 & 32 & -32 & -16 & 0 & 16 & 32 & -32 & 16 & 0 & 16 & 32 & -32 & -16 & 0 & 16 & 32 & -32 & -16 & 0 & 16 & 32 \\ 32 & 32 & 32 & 32 & 32 & -16 & -16 & -16 & -16 & -16 & 0 & 0 & 0 & 0 & 0 & 0 & 16 & 16 & 16 & 16 & 16 & 32 & 32 & 32 & 32 \end{bmatrix}$$

Function 6:

$$f_6(\mathbf{x}) = \sum_{i=1}^{n}(x_i^2 - 10\cos(2\pi x_i) + 10), -5.12 \le x_i \le 5.12$$

minimum:

Example: Consider a function $f(x,y,z) = x - 2xy + 3z$ to be minimised, where $2 \le x, y, z \le 5$. Each variable is represented by 2 bits.

Chromosome: $[x, y, z]$

After encod

Encoding\

Decoding: $x = x_{lo} + decimal(1001\cdots$

With $x_{lo} = 2$, $x_{hi} = 5$ and $m = 2$,

$00 \rightarrow 2 + decimal(00) \times \frac{5-2}{2^2-1} = 2$

$01 \rightarrow 2 + decimal(01) \times \frac{5-2}{2^2-1} = 3$

$10 \rightarrow 2 + decimal(10) \times \frac{5-2}{2^2-1} = 4$

$11 \rightarrow 2 + decimal(11) \times \frac{5-2}{2^2-1} = 5$

Example: Consider a function $f(x,y,z) = x - 2xy + 3z$ to be minimised, where $2 \le x, y, z \le 5$. Each variable is represented by 2 bits

Step 1: Pop

1	101110	4, 5, 4	
2	000011	2, 2, 5	
3	001100	2, 5, 2	
4	110100	5, 3, 2	-19

Example: Consider a function $f(x,y,z) = x - 2xy + 3z$ to be minimised, where $2 \le x,y,z \le 5$. Each variable is represented by 2 bits

Step 2: Ran

1	101110	4, 5, 4	
2	110100	5, 3, 2	
3	001100	2, 5, 2	
4	000011	2, 2, 5	9

Example: Consider a function $f(x,y,z) = x - 2xy + 3z$ to be minimised, where $2 \le x,y,z \le 5$. Each variable is represented by 2 bits.

Step 3: Selection with cost weighting (roulette wheel weighting)

n	Chro					P_n
1	101110	4, 5, 4				
2	110100	5, 3, 2	-19	-19	–	
3	001100	2, 5, 2	-12	-12	–	
4	00001	2, 2, 5	9			

$$P_n = \left|\frac{C_n}{\sum_{m=1}^{N_{keep}} C_m}\right| = \left|\frac{C_n}{\sum_{m=1}^{3} C_m}\right| = \left|\frac{C_n}{-33-28-21}\right| = \left|\frac{C_n}{-82}\right|$$

$$P_1 = \left|\frac{-33}{-82}\right|, P_2 = \left|\frac{-28}{-82}\right|; P_3 = \left|\frac{-21}{-82}\right|$$

- Generate two random numbers: 0.9649, 0.2785

What happen if the same chromosome is chosen?

Example: Consider a function $f(x,y,z) = x - 2xy + 3z$ to be minimised, where $2 \leq x, y, z \leq 5$. Each variable is represented by 2 bits

Step 4: Cro

p_3: 0011

p_1: 101110

- Generate randomly a crossover point: 2

$offspring_1$: 001110

$offspring_2$: 101100

Example: Consider a function $f(x,y,z) = x - 2xy + 3z$ to be minimised, where $2 \le x, y, z \le 5$. Each variable is represented by 2 bits

Step 5: Mut

Ranked p

#mutatio

row = [2 2 3 4]; column = [4 5 2 5].

n	Chromosome	Chromosome after mutation	Decod	st
1	101110	101110	4, 5, 4	-4
2	110100	110010	5, 2, 4	-3
3	001100	011100	3, 5, 2	-21
4	001110	001100	2, 5, 2	-12

Example: Consider a function $f(x,y,z) = x - 2xy + 3z$ to be minimised, where $2 \le x, y, z \le 5$. Each variable is represented by 2 bits

- The 1^{st}
- **Ranke**

	Chromosome	Decoded x, y, z	Cost
1	101110	4, 5, 4	
2	011100	3, 5, 2	
3	001100	2, 5, 2	
4	110010	5, 2, 4	-3

- Repeat steps 1 to 5 until stopping criteria have been met.

Why do GAs work? Schema Theorem

- **Binary** genetic algorithms are considered.

Schema Theorem (Michalewicz, 1992)

"Short, low-order, above-average schemata receive exponentially increasing trials in subsequ

KING'S College LONDON
University of London

- **A scheme:** a template representing a subset of binary strings using symbols '0', '1' and $\star$ (don't care symbol).

Example:

Schema

Schema

Schema $\star\star\star\star\star$ matches 2 strings: {00000, 00001, , 11110, 11111}.

Schema 11100 matches 1 string {11100}.

String 11100 is matched by 2^5 schemata: {1110 $\star 0$,

$\star\star\star\star\star$}.

King's College London, University of London

Schema properties:

1. **Order** of the schema S (denoted by $o(S)$): the number of fixed positions ('0' and '1' positions), i.e., the length of the template minus the number of don't care
2. **Defi** [illegible] etween the first and the last fixed positions.

Example:

$S_1 = \star\star\star001\star110 \Rightarrow o(S_1) = 6, \delta(S_1) = 1$

$S_2 = \star\star\star\star00\star\star0\star \Rightarrow o(S_2) = 3, \delta(S_2) = 9-5 = 4$

$S_3 = 11101\star\star001 \Rightarrow o(S_3) = 8, \delta(S_3) = 10-1 = 9$

KING'S College LONDON
University of London

Binary GAs:

1. select $P(t)$ based on $P(t-1)$
2. recombine $P(t)$
3. eval
4. $t \leftarrow$

Binary GA

- $N_{keep} = 0$.
- Selection: roulette wheel weighting, **cost w**
- Crossover: single-point crossover (crossover points are allowed any point in between the first and last bits).
- Mutation: uniform mutation (with the probability of mutation p_m for each bit).
- Elitism is not implemented.

$\xi(S,t)$: the number of strings in the population at generation t matched by a schema S.

Example: Consider the following population (**costs are normalised** and thus of all negative):

$$v_3 = 110111 \quad f$$

$$v_4 = 000110 \quad f$$

$$v_5 = 110001 \quad f(v_5) = -8$$

$$v_6 = 110011 \quad f(v_6) = -3$$

$S_1 = \star\star01\star\star, \qquad \xi(S_1,t) = 3$

$f(S,t)$: average cost of all strings in the population matched by the schema S.

Assuming that there are p strings $\{u_1, \cdots, u_p\}$ in the population matched by a schema S

Example:

$S_1 = \star\star01\star\star$ is matched by $\{v_1, v_3, v_4\} = \{$

$f(S_1,t) = \frac{f(v_1)+f(v_3)+f(v_4)}{3} = \frac{-15-6-1}{3} = -7.3$

Objective: Investigate the probability of survival of all schemata S $(S_1, \cdots, S_{2^{N_{bits}}})$ in the GA process (selection, crossover, mutation).

Selection: The probability of selecting the string v_i:

where $F(t) = \sum_{j=1} f(v_j)$.

1. The number of strings matched by schema
2. The average probability of a string matched by schema S to be selected: $\frac{f(S,t)}{F(t)}$.
3. The number of strings to be selected for recombination: N_{pop} $(N_{keep} = 0)$.

University of London

$\xi(S,t+1)$: the number of strings matched by S after the selection process.

$$\xi(S,t+1) = \xi(S,t)\frac{f(S,t)}{F(t)} N_{pop}$$

$$=$$

where $\overline{F}($

$$\xi(S,t+1) = \xi(S,t)\frac{f(S,t}{\overline{F}(t)}$$

$$= \xi(S,t)\frac{\overline{F}(t)}{\overline{F}(t)}$$

$$= \xi(S,t)(1+\varepsilon(t))$$

where $\varepsilon(t) = \frac{f(S,t)) - \overline{F}(t)}{\overline{F}(t)}$.

$$
\begin{aligned}
\xi(S,t+1) &= \xi(S,t)(1+\varepsilon(t)) \\
&= \underbrace{\xi(S,t-1)(1+\varepsilon(t-1))}_{\xi(S,t)}(1+\varepsilon(t)) \\
&= \xi(S,t-2)(1+\varepsilon(t-2))(1+\varepsilon(t-1))(1+\varepsilon(t)) \\
&\ \ \vdots \\
&= \xi(S,1)(1+\varepsilon(1))(1+\varepsilon(2))\cdots
\end{aligned}
$$

- $\varepsilon(t) > 0$ for most of t: $\xi(S,t+1)$ is increasing
- $\varepsilon(t) < 0$ for most of t: $\xi(S,t+1)$ is decreasing

Implication: Above average schemata receive increasing number of strings in the next generation; however, below average schemata will die out as t increases.

Recombination - Crossover

Probability of destruction of a schema S: $p_d(S) = \frac{\delta(S)}{N_{bits}-1}$

Probability of schema S survival: $p_s(S) = 1 - \frac{\delta(S)}{N_{bits}-1}$

Example

A string v

$S_1 = 110$

$\vdots$

$S_a = \star\star\star 111 \star\star\star\star$ $\quad p_d(S_a) =$

$\vdots$

$S_b = 11\star\star\star\star\star\star 10$ $\quad p_d(S_b) = \frac{9}{9} = 1, p_s(S_b) = 0$

$\vdots$

$S_{2^{N_{bits}}} = \star\star\star\star\star\star\star\star\star\star$ $\quad p_d(S_{2^{N_{bits}}}) = \frac{0}{9} = 0, p_s(S_{2^{N_{bits}}}) = 1$

King's College London
University of London

$v_1' =$ **1101110010**

$v_2' = 0100000011$

Example 1: After crossover

$o_1^c =$ **1101**

$o_2^c = 010$

S_a survives but not S_b.

Example 3: After crossover

ve.

Example 2: After crossover

$o_1^c =$ **110**$|0000011$

$o_2^c = 010|$**1110010**

S_a survives but not S_b.

Exa

$o_1^c =$ **11011**$|00011$

$o_2^c = 01000|$**10010**

Both S_a and S_b cannot survive.

King's College London
University of London

$$p_s(S) = 1 - \frac{\delta(S)}{N_{bits} - 1}$$

Modification of $p_s(S)$ considering that, e.g., the schema of v_1' and v_2' is the same

Schema growth equation with the consideratio

$$\xi(S, t+1) \geq \xi(S, t)\frac{f(S, t)}{\overline{F}(t)} \quad \frac{\quad}{bits -}$$

King's College London
University of London

Mutation: Uniform mutation - each bit will be mutated if a random number $r < p_m$, where $p_m \in [0, 1]$ is the probability of mutation

Probabilit

Probability of a schema S survival (no mutation tak

$$p_s(S) = (1 - p$$

Why do GAs work? Schema Theorem

Schema growth equation with the consideration of crossover and mutation:

$$\xi(S,t+1) \geq \xi(S,t)\frac{f(S,t)}{\overline{F}(t)}\left(1-\frac{\delta(S)}{N_{bits}-1}\right)(1-p_m)^{o(S)}$$

Schema T

"Short, low-order, above-average schemata receive

in subsequent generations of a genetic algorithm."